Issue Number 3 — July 1983

STOP PRESS DRAGON

EDITORIAL

Welcome to the third edition of Stop Press which, like its predecessors, has been sent to all those Dragon owners who returned their guarantee card Those who have not (and there are many) are not as yet registered members of Dragon Users Club and will not be on the mailing list if you know of anybody in this category, gently prod them to return their card so that future issues of Stop Press may be directed to them

Though not (as yet) a fully fledged magazine, Stop Press continues, in this and future issues, to offer Dragon owners pages of program-oriented material Programs which (a) do something interesting (b) illustrate programming techniques and (c) can be typed into Dragon in a short time, are ideal for inclusion in Stop Press Most of the material presented falls into at least two of these categories!

Two features in the last issue have their counterparts within. Indeed we hope that Machine Code Corner and the Young User Pages will be regular features. In the belief that there are many Dragon users who will appreciate a gentle introduction to machine code, MCC (nothing to do with cricket!) continues the series by showing how the commands introduced last time can be harnessed to perform high-speed, high-resolution graphics As each issue is published, readers will be able to accumulate sufficient expertise to make modest use of machine code in their programs

This does not deny the fact that Dragon's BASIC interpreter offers a rich vocabulary of commands for constructing programs, whether they be games or whatever Two of the commands, SOUND and PLAY, are explored in the YOUNG USER PAGES which also includes a puzzle and a competition

Mr W Harrold sent a contribution (Patterns) which exploits LINE He and others will be interested in 'Take 1ØØØ lines' where other intriguing aspects of the LINE command are investigated Incidentally the pictures in this and the previous issue were not produced by ruler and pencil but by Dragon and a Hewlett Packard graph plotter

Another contribution, this time from a young user in Cardiff, is a game of skill called 'Chicane' Full marks to Gareth Rowlands for a superbly designed game and a compact program We look forward to many other such contributions from young and old alike

The summer months may see a lull in programmers' activities as they take advantage of the hot weather! **But don't forget to send your entries for the 'Draw a Dragon Logo' competition (see Issue 2) by the end of July please, to the editorial address.**

Miss Cathy Hyde
Dragon Data Ltd Kenfig Industrial Estate
Margam Port Talbot West Glamorgan

As for Dragon Data, every month, hot weather or otherwise, is a month of intense activity in preparing for the launching of new ventures and the consoildation of software for Dragon 32 Stop Press will play an increasing role in providing up to date information in the months ahead

PRICE CHANGE

On the 23 May the retail price of Dragon 32 was reduced to £175 including V A T At the same time the following reductions were made Joysticks selection 2, Graphic Animator, examples from the manual, Dragon Mountain and Dragon Selection 1 down from £7 95 to £4 95, Ghost Attack cartridge down from £24 95 to £19 95 These price adjustments are to be enhanced by the release of approximately 25 new titles over the next few weeks with many more to follow shortly there after Also note that the Dragon disc drive and operating system was launched at the recent Computer Fair at Earls Court, and disc-based software will be released shortly

MACHINE CODE CORNER

An area of programming in which the effect of machine code is particularly striking is the animation of graphics. To see the limitations of BASIC in this respect, try keying in this short program

```
1Ø PMODE4 1 PCLS1 SCREEN1 Ø COLORØ 1
2Ø CIRCLE(1Ø 2Ø) 9
3Ø DIMA(11)
4Ø GET(Ø 1Ø) - (2Ø 3Ø) A G
5Ø FORI = 1TO235 PUT(I 1Ø) - (I+2Ø 3Ø) A PSET NEXTI
5Ø GOTO6Ø
```

A small circle is drawn on the left of the screen, and moved by the finest possible increments, to the right The "animation" takes about 15 seconds Our more artistic readers may like to convert the circle into a snail! (By the way, note how small the array dimension is – it needs to be about a fortieth of the area of the "GET" in pixels, which is 21x21 = 441.)

Well, how much can this be improved by using machine code? Try this one

```
1Ø DATA  1Ø 8E 7 4Ø 86 8 3Ø A4 C6 15 66 84 66 1
    66 2 66 3
2Ø DATA  3Ø 88 2Ø 5A 26 F2 4A 26 EB 31 21 1Ø  8C 7 5D,26 E1 39
3Ø FORI = ØTO35 READX$ POKEI+32ØØØ VAL( "&H  +X$)
    NEXT
4Ø PMODE4 1 PCLS1 SCREEN1 Ø COLORØ 1
5Ø CIRCLE(1Ø 2Ø) 9
6Ø EXEC32ØØØ
2Ø GOTO7Ø
```

This, of course, is really two short programs Lines 1Ø–3Ø POKE in the machine code and lines 4Ø–7Ø execute it Even with the READ POKE loop the program is finished in one second The action itself takes one fifth of a second After the first RUN, the machine code will be in position, so RUN4Ø will give you an action replay

This is obviously faster than you will need for most purposes, and can be slowed by the use of delay loops As with the BASIC program, every intermediate position is occupied, so a completely smooth movement is possible at any speed The BASIC can be speeded up by leaving out some of the intermediate positions but this results in a jerky movement

Before we look in detail at animations of this sort, we must investigate the means by which Dragon stores graphic information We shall concentrate on PMODE4,1 This makes use of memories $6ØØ to $1DFF (or 1536 to 7679) i.e a total of 6144 bytes In this mode, each of the 256x192 pixels is either 'on (green or buff) or 'off' (black) If it is 'on' it has value 1, if it is off it has value Ø Pixels are grouped together in sets of eight so the top line of the screen is made up of 32 sets of 8 pixels, and each set is represented by one byte of memory For example, if the first eight pixels are on, on, off, off, on, on, off, off', then $6ØØ will contain the binary number 11ØØ11ØØ (decimal 2Ø4) This can be seen by running

```
1Ø PMODE4 SCREEN1 Ø PCLS1 POKE&H6ØØ 2Ø4
2Ø GOTO2Ø
```

(PCLS1 turns the whole graphics screen on i.e. it places binary 11111111, or decimal 255, in each memory, $6ØØ to $7DFF.) So the rows of the screen are contained in memories $6ØØ–$61F, $62Ø–$63F, $64Ø–$65F etc

From this discussion, you may have concluded that it is easier to move things up and down than sideways – and you would be right The byte configuration remains the same in vertical movement So let's see if we can make the balloon go up, in a controlled sort of way

Let's start in BASIC

```
1Ø PMODE4 1 PCLS1 SCREEN1 Ø COLORØ 1
2Ø CIRCLE(128 12Ø) 7 PAINT(128 17Ø)
3Ø LINE(121,17Ø) - (128 186),
    PSET LINE(135 17Ø) - (128 166) PSET
4Ø LINE(124 166) - (132 19Ø) PSET B
5Ø DIMA(12) GET(12Ø 162) - (136 191) A G
6Ø FORI = 1TO162 PUT(12Ø 162-I)
    - (136 191 - I) A,PSET NEXT
7Ø GOTO7Ø
```

Our balloon drawing is (conveniently) confined to the 16th and 17th bytes in rows 163 to 191 As with GET and PUT it is useful to include the bottom blank row to avoid the problem of "rubbing out" the bottom row of the picture as it moves up Row 163 starts at byte $6ØØ + 32* 163 = $1A6Ø, so the top of the picture is $1A6F and $1A7Ø then $1A8F and $1A9Ø etc

The following assembly language routine will do the job

```
                                           Machine Code
 1              LDY         #$1A6F         1Ø 8E 1A 6F
 2  LOOP1       LEAX        Y              3Ø A4
 3  LOOP2       LDD         ,X             EC 84
 4              STD         -32 X          ED 88 EØ
 5              LEAX        32 X           3Ø 66 2Ø
 6              CMPD        #$FFFF         1Ø 83 FF FF
 7              BNE         LOOP2          26 F2
 8              LEAY        -32 Y          31 A8 EØ
 9              CMPY        #$6ØF          1Ø 8C Ø6 ØF
1Ø              BNE         LOOP1          26 E7
11              RTS                        39
```

One command you may not have met before is Load Effective Address (LEA) The statement LEAX ,Y may be compared with LDX ,Y but instead of loading into X the data *indexed* by Y the actual *address* (the "value" of Y) is loaded into X The opcodes for LEAX and LEAY are respectively, hex 3Ø and hex 31

The numbers in the *operand* field prior to the comma in lines 4, 5 and 8 are called *offsets* They cause the *command* to be applied to the register value + offset In other words, line 4 stores the value of D in the memory whose address is 32 less than the value in X, line 5 *increments* X by 32 line 8 *decrements* Y by 32

The byte following an opcode which requires a register as part of its operand is called a *postbyte* We met one in the last edition ,X+ is represented by postbyte $8Ø For our present program, we need to know that ,X and ,Y are respectively $84 and $A4 (with no offset) and with offsets in the range 16 to 127 or –128 to –17 the postbytes are $88 and $A8 (for X and Y), followed by another byte – the offset Negative offsets are FF= –1, FE= –2, etc

Our BASIC program, incorporating the machine code, is

```
1Ø DATA  1Ø 8E 1A 8F 3Ø A4 EC 84 EO 88 E8
   3Ø 88 26 1Ø 83 FF FF
2Ø DATA  26 F2 31 A8 EØ 1Ø 8C Ø F 2Ø,E7 39
3Ø FORI–ØTD2Ø READX$ PDKE32ØØØ
   +I,VAL('&H' +X$) NEXT
4Ø PMODE4 1 PCLS1 SCREEN1 Ø:CDLDRØ 1
5Ø CIRCLE(128 17Ø) 7 PAINT(128 17Ø)
6Ø LINE(121 17Ø)–(128 186) PSET
   LINE(13Ø 17Ø) (128 18Ø) PSET
7Ø LINE(124 186)–(132 18Ø) PSET B
1ØØ EXEC32ØØØ
11Ø GOTO11Ø
```

Finally, to control the speed, we insert between lines 7 and 8 the following delay loop

```
                                 Machine Code
            LDX       $7FFE      BE 7F FE
LOOP3       LEAX      –1 X       3Ø 1F
            BNE       LOOP3      26 FC
```

The postbyte of LEAX –1,X requires explanation When the offset lies between –16 and 15, the postbyte represents the offset For offsets of Ø to 15, postbyte = offset For offsets of –18 to –1 postbyte = offset + $2Ø

The only other modification to the machine code is to the relative address of line 1Ø 26 E7 becomes 26 EØ The following program gives a controlled movement:

```
1Ø DATA  1Ø 8E 1A 8F 3Ø A4 EC 84 EO 88 EØ
   3Ø 88 2Ø 1Ø 83 FF FF
2Ø DATA  26 F2 BE 7F FE,3Ø 1F,26 FC,31 A8
   EØ 1Ø 8C Ø F,26 EØ 39
3Ø FORI–ØTO36 READX$ POKE32ØØØ+I VAL('&
   H' +X$) NEXT
4Ø PMODE4 1 PCLS1 SCREEN1 Ø COLDRØ 1
5Ø CIRCLE(128 17Ø) 7 PAINT(128 17Ø)
6Ø LINE(121 17Ø)–(128 186) PSET
   LINE(13Ø 17Ø)–(128 18Ø) PSET
7Ø LINE(124 186)–(132 18Ø) PSET B
8Ø POKE&H7FFE,5 POKE&H7FFF 255
1ØØ EXEC32ØØØ
11Ø GDTO11Ø
```

Line 8Ø puts the value 5*256+255 = 1535 into memory $7FFE/$7FFF Smaller values give a faster movement larger ones give a slower movement The result – a very fine control over the speed of the balloon

PCOPY with Care!

When we write BASIC programs on DRAGON 32 we are well protected by a sophisticated error trap which prevents us from giving commands that cannot be carried out As an example of this, turn on your Dragon and type PMODE4,5 The answer is FC ERROR – you can t address page 5 in PMODE4 until you have PCLEARed 8 pages (the default is 4) So type PCLEAR8 and then PMODE4 5 This is accepted

Now type PCLEAR4 Back comes the answer FC ERROR You can't PCLEAR down to 4 pages while you are addressing page 5 So type PMODE4 1 and then PCLEAR4 This is accepted

Most commands are protected in this way The obvious exception is the POKE command – if you POKE addresses carelessly you are likely to lose control – try typing POKE136 Ø (136 and 137 control the cursor position)

A less obvious problem is the PCOPY command You might expect this to be protected in the same way as PMODE and PCLEAR – but it isn t If you type a program in, and then (without PCLEARING) type PCOPY 1 TO 5, your program will disappear This is because graphics page 1 has been copied to the location which held your program Admittedly this would be a silly thing to do but the problem is more likely to arise with programs containing statements like PCOPY I TO J, when it is quite easy to get the value of J wrong So PCOPY with care!

PATTERNS

Thanks to Mr William Harrold of Cambridge for this short program He suggests using densities of between 3 and 6 It is a program that we have seen around in vanous forms – the intereference patterns caused by the discrete nature of the screen pixels are generally referred to aa MOIRE patterns They will occur when lines are plotted cloae together

```
1Ø  ' PATTERNS
2Ø  ' WILLIAM HARROLD 1983
3Ø CLS INPUT ENTER DENSITY S
4Ø PMDDE4,1 SCREEN1 1 PCLS
5Ø FDR I=Ø TO 255 STEP S
6Ø LINE(I Ø) (256–I 191) PSET
7Ø IF I<192 THEN LINE(Ø I)–(256 191–I) PSET
8Ø NEXTI
9Ø A$–INKEY$ IF A$–"" THEN 9Ø ELSE 3Ø
```

PARTIAL RESTORE

When a program runs on a large number of DATA statements which may heva to be eccessed in any ordar the most convenient method to adopt is to read the whole dataset into an array and use an indexing variable to access it This takes up a fair amount of room in the machine, however, and an alternative is to read through the whole dataset avery time access is required, using the RESTORE command to raturn to the beginning of the list This is undoubtedly tha most efficient method of conserving mamory but is a rathar cumbersome procedura A partial RESTORE can ba achieved using tha DATA memory pointer in addresses $33 and $34 (dacimal 51/52) By PEEKing thasa addresses at suitable points the first time the data are read, we can store relevant values so that the pointer can be reset to those points by POKEing at e later time The principle is illustrated by this small program

```
 1Ø FORI–1TO3 X(I)–PEEK(51) Y(I)–PEEK(52)
 7Ø READX$ NEXT
 3Ø CLS PRINT@7 'WHICH  STATEMENT?'
 4Ø X$=INKEY$ IFVAL(X$)=ØORVAL(X$)
    >3THEN4Ø
 5Ø X=VAL(X$) POKE51 X(X) POKE52,Y(X)
 6Ø REAOX$ PRINT@71 X$
 7Ø FORI=ØTO1ØØØ NEXT GOTO3Ø
 8Ø OATA  FIRST STATEMENT
 9Ø OATA  SECOND STATEMENT'
1ØØ OATA 'THIRD STATEMENT
```

JOYSTICK GAME

This game is for two players (one on aach joystick) The laft joystick controls tha "L" and the right joystick controls the "R" The idea is to manoeuvre your letter about the scraen "running down" as many of the numbers as possible before your opponent gets to them You score the velue of the number and your score is shown at the top of the screen If you disappear off one edga of the screen, you will reappear immediately at tha opposite edge The numbers are 1–9 (so if you see an '87', it is just an eight and a seven togethar) New numbers appear from time to tima to keep you busy The time limit is controlled by the 499 in line 8Ø.

```
 1Ø REM JOYSTICK GAME
 2Ø REM A O  MAYER 1983
 3Ø CLS4 GOSUB21Ø
 4Ø L(1)–12 GOSUB23Ø J(1)=J+1Ø55
    POKEJ(1),L(1)
 5Ø L(Ø)=1Ø GOSUB23Ø J(Ø)=J+1Ø55
    POKEJ(Ø) L(Ø)
 6Ø PRINT'LEFT=Ø" PRINT@1Ø "RIGHT=Ø
    ' PRINT@21 ' TIME–Ø",
 7Ø S(Ø)–Ø S(1)=Ø S=1
 8Ø FORT=1TO499
 9Ø IFINT(T/1ØØ)*1ØØ–T ORS(Ø)+S(1)=S*45
    THENGOSUB21Ø
1ØØ JY–RNO(2)–1 GOSUB24Ø
11Ø JY=1–JY GOSUB24Ø
12Ø PRINT@5 S(1), PRINT@16 S(Ø) PRINT@26 T,
13Ø NEXTT
14Ø IFS(Ø)>S(1)THENPRINT@263 "RIGHT IS THE
WINNER" GOTO17Ø
15Ø IFS(1)>S(Ø)THENPRINT@263  LEFT IS THE
WINNER" GOTO17Ø
16Ø PRINT@26Ø 'THE RESULT IS A ORAW'
17Ø PRINT@324 "DO YOU WANT ANOTHER GAME?",
18Ø X$=INKEY$ IFX$="" 'THEN18Ø
19Ø IFX$=" Y 'THEN3Ø
2ØØ IFX$=" N 'THENSTOPELSEGOTO18Ø
21Ø FORI=1TO9 GOSUB23Ø
22Ø POKE1Ø55+J I+112 NEXT S=S+1 RETURN
23Ø J=RND(24Ø)*RND(2) IFPEEK
    (1Ø55+J)><191THEN23ØELSERETURN
24Ø H=Ø R=JOYSTK(2*JY) IFR<16THENH=–1
25Ø IFR>48THENH=1
26Ø V=Ø R=JOYSTK(2*JY+1)
    IFR<15THENV=–32
27Ø IFR>48THENV=32
28Ø X=J(JY)+H+V IFX>1535THENX=X–48Ø
29Ø IFX<1Ø56THENX=X+48Ø
3ØØ IFX–J(1–JY)ORX=J(JY)THENRETURN
31Ø IFPEEK(X)><191THENS(JY)=
    S(JY)+PEEK(X)–112
32Ø POKEJ(JY) 191 POKEX L(JY) J(JY)=X
    RETURN
```

The program uses the JOYSTK function to check the positions of the joysticks Soma versions of the Dragon primer give the JOYSTK valuae the wrong way round The corract varsion is

JOYSTK(Ø)	Right	Horizontal
JOYSTK(1)	Right	Vertical
JOYSTK(2)	Left	Horizontal
JOYSTK(3)	Left	Vertical

If you want to usa the joysticks in a machine code program, the procadure ie as follows Usa the JSR instruction to execute the ROM subroutine which begins at address $8Ø12 (decimal 32786), then the relevant velues, for tha moment the subroutine was executed, will appear in eddreesas $15A $15B $15C and $15D (decimal 346–349) This method can, of course ba usad from BASIC, using EXEC These two programs are equivalent try them with your joysticks

```
(a) 1Ø PRINT JOYSTK(Ø) JOYSTK(1) JOYSTK(2)
    JOYSTK(3) GOTO1Ø
(b) 1Ø EXEC32786 PRINTPEEK(346) PEEK(347) PEEK(348),
    PEEK(349) GOTO1Ø
```

s for your 32

CHICANE

This driving game was sent to us by Gareth Rowlands of Cardiff. A width of road between 2 and 7 may be selected (we suggest you start at 7!) and the idea is to stay on the road for as long as possible, using the left and right arrow keys to steer. Those of us who were brought up in a more sedentary age may benefit by editing line 18Ø, to adjust the upper limit from 7 to 9. The benefits of selecting easy options do not last long – the road narrows as you progress.

```
Ø REM **chicane** G ROWLANDS
1Ø CLS CLEAR2ØØ 29999 GOSUB 23Ø
2Ø AUDIDDFF MOTOROFF M1$=
   'T6D1CEGECCCEDEGEC GOTO 17Ø
3Ø FOR I=31968 TD P LD POKE I,255 NEXT
4Ø FOR I=P+1 LD TD P+RD POKE I 175 NEXT
5Ø FOR I=P+1+RO TO 31999 POKE I 255 NEXT RETURN
6Ø FDRI–1 TO 16 SDUNDI*1Ø+25 1 EXEC32Ø51 NEXT TIMER=Ø
7Ø FDR D=1 TO 1Ø
8Ø FDR I=1 TO 16 R=RND(2) IF R=1 THEN LC 175 RC=255
   ELSE LC=255 RC–175
9Ø CP=CP+(PEEK(343)=223)–(PEEK(344)=223)
   POKEP–LD LC POKEP+RD RC P=P+(P≥31974 AND R=
   1)–(P<31993 AND R=2) EXEC32Ø51 IF PEEK(CP)=255 THEN
   15Ø ELSE POKECP 3Ø NEXT
1ØØ IF P<31984 THEN D=2Ø ELSE D=Ø
11Ø PRINT@D INT((TIMER+1ØØ)/1ØØ)*5
   PRINT@D+32 MILES' SCREEN Ø 1 NEXT
12Ø A=A+(A>2) SDUND2ØØ 1
13Ø PRINT@D+64 ROAD PRINT@O+96 NARROWS
14Ø LD=Ø RD=Ø B=A GOSUB 21Ø GOSUB 3Ø GDTD 7Ø
15Ø T=TIMER FOR I=1 TD 17 EXEC32Ø51
   POKECP+RND(3)–2 239 POKECP+RND(3)–2,2Ø7
   POKECP+RND(3)–2 96 PDKECP+RND(3)–2 159 NEXT
155 IFT<ØTHENT–Ø
16Ø PLAY M1$ PRINT@1ØØ YDU
   SURVIVEO 'INT(T/1ØØ)*5 'MILES'
17Ø PRINT@23Ø WIDTH OF RDAD? ' SCREENØ 1
18Ø A$=INKEY$ IF VAL(A$)<2 OR VAL (A$)>7 THEN 18Ø
19Ø CLSØ LD=Ø RD=Ø A=VAL(A$) B=A.
   GOSUB 21Ø
2ØØ P=31982 CP=151Ø GDSUB 3Ø GOTO 6Ø
21Ø B=B–1 LD=LO+1 IF B<1 THEN RETURN
22Ø B–B–1 RD=RD+1 IF B<1 THEN RETURN ELSE 21Ø
23Ø P=P+1 READ R$ IF R$= END THEN RETURN
24Ø POKE P+32Ø5Ø VAL( &H +R$) GDTO 23Ø
25Ø DATA 8E Ø5 EØ 1Ø 8E
26Ø DATA Ø6 ØØ EC 83 ED
27Ø OATA A3 8C Ø4 ØØ 2E
28Ø DATA F7 8E 7D ØØ EC
29Ø DATA 83 ED A3 1Ø 8C
3ØØ DATA Ø4 ØØ 2E F6 39
31Ø DATA END
```

The machine code (lines 25Ø to 3ØØ) controls the down scroll. The following listing may be useful

```
                              Machine Code
        LDX     #$5EØ         8E Ø5 EØ
        LDY     #$6ØØ         1Ø 8E Ø6 ØØ
LOOP    LDD      X            EC 83
        STD      Y            ED A3
        CMPX    #$4ØØ         8C Ø4 ØØ
        BGT     LOOP          2E F7
        LDX     #$7DØØ        8E 7D ØØ
LOOP2   LDD      X            EC 83
        STD      Y            ED A3
        CMPY    #$4ØØ         1Ø 8C Ø4 ØØ
        BGT     LOOP2         2E F6
        RTS                   39
```

CODEBREAKER

This is a short program to further illustrate the power of Dragon's string handling capabilities. It translates a typed message into a code which is constructed by randomly permuting the allowed letters. Because the random permutation has been generated using RND(–1) to initialise the random sequence, the code can be deciphered using the same program providing the transmission code is known.

Of course, given time all such codes can be 'cracked' and one of the ways to do that is to use the knowledge gained from 'Letter Count' concerning the frequencies of each of the letters of the alphabet. For a literary reference read 'THE GOLD BUG' in Edgar Alan Poe's Tales of Mystery and Suspense.

```
1 REM COOEBREAKER A M SYKES MAY83
1Ø CLS CLEAR1ØØØ CH$=
   ABCDEFGHIJKLMNDPDRSTUVWXYZ, ? –
2Ø CC$=CHR$(13)+CHR$(8)
3Ø INPUT' 1 FOR CDDING 2 FOR OECODING C
4Ø ON C GDSUB 5Ø 11Ø STOP
5Ø R=RND(1ØØ) PRINT YDUR TRANSMISSION
   CODE IS R
6Ø GDSUB13Ø
7Ø PRINT@128 ' NOW KEY IN YDUR MESSAGE
8Ø PRINT (<=255 CHARACTERS LONG AND ENDING
   WITH <ENTER>)'
9Ø GOSUB24Ø GDSUB16Ø CLS
1ØØ PRINT@1 YOUR MESSAGE READS
   GOSUB24Ø PRINT@128 CM$ RETURN
11Ø INPUT ENTER TRANSMISSION CODE R
12Ø GDTD 6Ø
13Ø Z=RND(–1) FOR I=1 TO R Z=RND(3Ø) NEXTI
14Ø CD$=CH$ FOR I=3Ø TD 1 STEP –1
15Ø Z=RND(I) CD$=LEFT$(CD$ Z–1)+RIGHT$(CD$ 3Ø–Z)
   +MID$(CD$ Z,1) NEXTI RETURN
16Ø M$= CM$= '
17Ø K$=INKEY$ IF K$= THEN 17Ø
18Ø CH–INSTR(1 CH$ K$) CC=INSTR(1 CC$ K$) IF CH=Ø AND
   CC=Ø THEN 17Ø
19Ø IF CC=1 THEN RETURN
2ØØ IF CC=2 THEN L=LEN(M$) M$–LEFT$(M$ L–1)
   CM$–LEFT$(CM$ L–1) PRINT@1 M$ GOTO17Ø
21Ø M$=M$+K$ PRINT@1 M$ DN C GOSUB22Ø 23Ø GOTO17Ø
22Ø K=INSTR(1,CH$ K$) CM$=CM$+MID$(CD$ K 1) RETURN
23Ø K=INSTR(1,CD$ K$) CM$–CM$+MID$(CH$ K,1) RETURN
24Ø FOR J=1 TD 2ØØØ NEXTJ CLS RETURN
```

LETTER COUNT

This program counts the number of occurrences of letters in writing Just type in the words and watch the bar chart build up There is no backspace so type carefully The program stops when any bar reaches the top Then you can press ENTER for a table of the number of occurrences of each letter You can press ENTER for the table at any time during a run but the program is then terminated

```
  1 REM LETTER CDUNT M PEARSDN MAY 83
 10 CLS0 DIMB(25)
 20 PRINT@483 ABCDEFGHIJKLMNDPQRSTUVWXYZ
 30 PRINT@384  5-  PRINT@224 15-  PRINT@64 25-
 40 PRINT@1 A$ K$=INKEY$ IF K$="" OR K$=CHR$(8) THEN
    40
 50 IF A<> -52 THEN 70
 60 CLS FDR I=0 TO25 PRINT
    TAB(5) CHR$(I+65) B(I) NEXT END
 70 A$=A$+K$ A=ASC(K$)-65
 80 L=LEN(A$) IF L>29 THEN A$=MID$(A$,L-28)
 90 IF A<0ORA>65 THEN40 ELSE B(A)=B(A)+1
100 I=INT(B(A)/2) IF B(A)/2=I THEN C=223 I=I-1 ELSE
    C=211
110 C=C+32*(A/2-INT(A/2))
120 PRINT@A+3+32*(14-I) CHR$(C)
130 IF B(A)<28 DR INKEY$=CHR$(13) THEN 40 ELSE 130
```

Another way to display the bar chart requires the following changed lines with line 110 deleted

```
 10 CLS DIMB(25)
100 I=INT(B(A)/2) IF B(A)/2=I THEN C=58 I=I-1 ELSE C-46
```

TAKE 1000 LINES

The LINE command can draw lines on the screen in any direction Unlike the DRAW command it is not limited to eight directions The command requires the co-ordinates of the end points of the line It is very useful to have the high-resolution grid for reference Once the end points have been established the LINE command takes another parameter, PSET or PRESET PSET draws the line in the foreground colour whereas PRESET draws it in the background colour The colours can be chosen using the COLOR command This has two paremeters FG, the foreground colour and BG, the background The colours must be chosen with respect to the SCREEN in use If the COLOR command is omitted the foreground colour will be the highest number available on that screen and the background the lowest There is another form of the command LINE-(XB,YB) which draws a line between the new point and the last one used in a LINE command If the first LINE command in the program is of this form, e line is drawn from the centre of the screen to the point Using PRESET, the first time round might be one wey of removing this line In addition to PSET and PRESET there is another parameter by edding B to the LINE command a box is drawn instead of the line, the diegonal of the box being the originel line If BF is added the box is filled in the foreground colour

The following program shows a triangle subroutine in use The co ordinates of the three points are chosen at random and the colours are rotated To colour the triangles in, a point inside the triangle must be found for the PAINT command In this case the points of the triangle are chosen at random So which point lies inside the triangle? The point whose co-ordinetes are the average of the X values, (XA+XB+XC)/3 and the average of the Y values (YA+YB+YC)/3 will certainly lie inside

```
 1 REM TRIANGLES M PEARSDN MAY 83
10 PMDDE3 SCREEN1 0 PCLS
20 FOR C=2 TD 4 CDLDR C 1
30 XA=RND(255) XB=RND(255) XC=RND(255)
40 YA=RND(191) YB=RND(191) YC=RND(191)
50 GOSUB70 PAINT((XA+XB+XC)/3
   (YA+YB+YC)/3) C C NEXT
60 GDTD60
70 LINE(XA YA)-(XB YB) PSET
80 LINE-(XC YC) PSET LINE-(XA YA)
   PSET RETURN
```

Writing a lot of LINE commands can be tedious One way out is to write the command es a subroutine and change the values of the end points either by defining them directly in the program e g XA=100, or, by reading them from DATA statements

The following demonstration program shows how DATA statements can be used with a LINE subroutine to make a picture out of filled boxes. The COLOR command allows the boxes to be different colours. Only when a variable is changed does it's new value have to be READ. The program draws ten boxes and then RUN should reveal one very familiar 'box'.

```
 1REM BOX M PEARSON MAY 83
 10 PMODE3 SCREEN1 1 PCLS C1=7 C2=8 CO=C1
 20 DEF  FNC(C)=-(C=C1)*C2-(C=C2)*C1
 30 FOR I=1 TO 5 READ XA YA XB YB C GOSUB1000 NEXT
 40 FOR I=1 TO 2 READ YA YB GOSUB1000 NEXT
 50 READ  XA XB GOSUB1000
 60 FOR I=1 TO 2 READ YA YB GOSUB1000 NEXT

 120 GOTO120
1000 COLOR  C 5 LINE(XA YA)-(XB YB) PSET BF RETURN
2000 DATA  20 90 190 170 8 90 90 135 130
     6 60 150 104 144 7
2010 DATA  106 110 170 180 7 148 60 165 70 7
2020 DATA  75 85 90 100 163 170 75 85 80 70
```

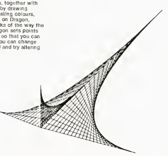
Another way to change the co ordinates is in a FOR ... NEXT loop. This works well when the coordinates change in a regular way. The next few lines use this method to shade the top and sides of the 'box'. Five LINE commands, together with PAINT could have been used but by drawing lines very close together in alternating colours, new colours, not readily available on Dragon, appear. These are created as quirks of the way the eye combines colour and the Dragon sets points. The colours are defined in line 10 so that you can experiment with different ones. You can change the step length in lines 70 and 100 and try altering the SCREEN.

Add these lines to the previous program

```
 70 FOR I=1 TO 30 STEP2 COLOR  CO LINE(180+I 90-I)-(180
    +I 171-I) PSET
 80 CO=FNC(CO) NEXT
100 FOR I=1 TO 30 COLOR CO LINE (20+I 90-I)-
    (180+I 40-I) PSET
110 CO=FNC(CO) NEXT
```

There is another sort of shading available with a LINE command. This involves joining points along two lines. Each line is divided into many equal intervals and the points are joined as in peg and silk designs. The effect is of a moulded shape. The following program shows this beautifully.

```
 1 REM  CONCORDE  A M SYKES
10 PMODE4 1 COLOR0 5 SCREEN1 1 PCLS5
20 FOR J=1 TO 3
30 READ N A B C D AA BB CC DD FOR I=0 TO N
40 Z=I/N X=A+(C-A)*Z Y=B+(D-B)*Z U=AA
   +(CC-AA)*Z V=BB+(DD-BB)*Z
50 LINE(X Y)-(U V) PSET NEXTI J
60 GOTO60
70 DATA  25 12 12 180 100 180 150 80 180
80 DATA  15 12 12 180 150 180 150 240 130
90 DATA  25 120 100 180 190 180 150 180 180
```



YOUNG USER PAGES

NINE-NINE-NINE

Does anyone know what sound a dragon makes? Dogs bark, cows moo, and E T says 'goohm', but what do dragons do? The sort of dragons that live in lairs and terrorise villagers by burning down houses whenever they cough that is Of course the sort that live in houses near the television only terrify the old folks (Over 3Ø s) We know these Dragons sing, make crunch noises and bullet whines But you haven't really got a well-trained Dragon until it makes noises that you want it to make You can use the SOUND command to tell your Dragon to make noises Type in SOUND 89 16 followed by ENTER You should hear a middle C lasting about a second If you don t then try turning up the sound or adjusting the tuning on your TV !

Try using different numbers The first determines the pitch and can be from 1 to 255 The second number controls the length of the note and must be greater than Ø

If you want to hear all possible pitches you can put the SOUND command into a loop
Going up

```
1Ø FOR P=1 TO 255
2Ø SOUNO P 1
3Ø  NEXT
RUN
```

The loop takes P through all tho values from 1 to 255 increasing one at a time We can change the loop to take P through all values from 255 to 1 decreasing 1 at a time Just change line 1Ø
Coming down

```
1Ø FOR P= 255 TO 1 STEP -1
```

Perhaps you want to play the same note over and over again Try

```
1Ø FOR J= 1 TO 5
2Ø SOUNO 17Ø 6
3Ø  NEXT
```

This loop takes J through values 1 to 5 but J isn't mentioned inside the loop so the program just does the same thing five times How dull! Let's change line 2Ø to

```
2Ø SOUNO 17Ø 6 SOUND 15Ø 6
```

Run this Does it remind you of anything? Now add these lines to give you the familiar drop in pitch

```
4Ø SOUNO 169 6 SOUND 149 6
5Ø FOR J=1 TO 3
5Ø SOUNO 168 6 SOUND 148 6
7Ø  NEXT
```

You can make very small changes in pitch using the SOUND command, but you need a very good ear to decide which number gives which note However with the PLAY command you use letters CDEFGAB for a scale together with sharps (♯) and flats (-), or alternatively numbers separated by semi colons 1,2 3,4, 11,12 to play successive semitones This is the way to make your Dragon tuneful There are lots of instructions available and page 113 of the Dragon Primer lists them all

To make the siren sound using the PLAY command I decided to use B and B flat above middle C (Ø3) played very rapidly (T25Ø) to give the chord effect I played them 15 times to get the right length of note For the lower note I used F and F♯, but you can change the notes to suit yourself I wanted the siren to start quietly and come nearer so the volume had to be increased I could write lots if strings
PLAY"V1" + ,PLAY" V2" + and so on That's tedious so the changing numbers are in a loop For example lines 1Ø, 3Ø and 9Ø in the following program are

```
1Ø FOR L=1 TO 1Ø
3Ø PLAY 'V' +STR$(L)
9Ø NEXT L
```

(L for loudness)

When you use several loops in a program it's wise to state which variable you want when you end the loop with NEXT Hence NEXT L

Here is the program for a siren getting nearer

```
1Ø FOR L=1 TO 1Ø
2Ø FOR I=1 TO 15
3Ø PLAY  'V' +STR$(L)
4Ø PLAY 'T25Ø O3BB- '
5Ø NEXT  I
6Ø FOR I=1 TO 15
7Ø PLAY FF♯"
8Ø NEXT  I
9Ø NEXT  L
```

If you want to carry on and make the siren go past use

```
100 FOR L=20 TO 1 STEP -1
```

and a drop in pitch say A and B flat with F and E

Now a challenge! Can your Dragon ring like a telephone? If it can send your program to us and will reward the best entry with free Dragon software! For details, see Young Users Competition

Now follows a tiny program to develop weird sounds by changing one line

```
10 P$="T20ABBCA"
20 PLAY P$ GO TO 20
```

You must use the BREAK key to stop this program Here are some other Ideas for line 10

```
10 P$-"T200  ABBCA"
10 P$="T200  ABBP1'
10 P$="T2000  1ABBP1P1'
10 P$="T100D  1ABBP1P1
10 P$="T10005  ABBP1P1'
10 P$-"T100DGCCDEFGFGCCDEFG2"  (spacecraft sound)
10 P$="T100EDCCEDCCGFEEGFEE"  (mice running fast)
10 P$-"T10EDCCEDCCGFEEGFEE"  (slower mice)
10 P$="T2EDCCEDCCGFEEGFEE"  (sedate mice)
10 P$_"T100GFEDC"  (spaceship hit sound)
```

So PLAY AWAY!

COMPETITION
WIN 4 SOFTWARE CASSETTES

WIN 4 SOFTWARE CASSETTES (COMPETITION)

If you are twelve years old or under and would like to win some Dragon Software then devise a program on your Dragon 32 that uses the computer s SOUND or PLAY commands to mimic a telephone ringing The program should not be more than twenty lines long

The young programmer judged to have devised the neatest program will win 4 cassettes of their choice from the software listed on the back page and some posters illustrating some popular Dragon games

Send your entries to The Editor of the Newsletter at Dragon Data by 31st August 1983 The editorial decision is final

The editor, The Newsletter, c/o Dragon Data Ltd,
Kenfig Industrial Estate, Margam, Port Talbot, West Glamorgan

DRAGON PUZZLE 2

This program plays a tune but first you must get the lines in the correct order On the right are clues to the missing line numbers Put the answers in the spaces left for them and type RUN I'm sure you will recognise the tune

```
1 'DRAGON PUZZLE 2
_ PLAY"L8EEDL4 CL8C"
_PLAY C$FOR I=1TO2
_ PLAY"L4GECEL4 D"
_ H$="L4 G" F$="L4 F"
_ PLAY"L8EFL4GECDL4 C"
_ PLAY"L4EGL8G"+F$
_ PLAY H$+C$ NEXT
_Y$="L4GECD" C$="L8CEF"
```

_ saw the start of World War 1

_ is quite a score

_ is the year Columbus discovered the Bahamas

_ is a baker s dozen

_ is the year of the Dragon

_ and World War 1 ended

_, the year Herold saw no more

_ coming of age

SOLUTION TO DRAGON PUZZLE 1

```
10 CLS PRINT"DRAGON PUZZLE"
20 PRINT@294,"FIRE"
30 PRINT@263"GAMES"
40 PRINT@132"PIGLET"
50 PRINT@196"FOOTBALL"
60 PRINT@167"PYTHON"
70 PRINT@325"PANDA"
80 PRINT@230"GEORGE"
90 PRINT@101"BRAKE"
95 FOR I=1 TO 3000 NEXT
100 FOR I=3 TO 10 P=PEEK(1032+32*I)
110   PRINT@32*I+6 CHR$(P+32),
 NEXT PRINT@350
```

(See YOUNG USERS PAGE in previous issue)

NEW RELEASES from the Dragon's Lair

Here are 24 new titles in the official Dragon Data Software range that should be appearing in the shops about now Watch out for further Software news in STOP PRESS

M30526 STALAO ENO

No joysticks Two adventure games In stalag your goal is to escape alive, from a prisoner of war camp before it is bombed In Eno, you are searching for the money left by your late aunt To prove that you deserve the fortune that has been left to you, your aunt has hidden the cash

M30524 MANSION OF DOOM

No joysticks Mansion of doom is an adventure game in which you have been chosen to reunite the crown princess Marlena with the townspeople of her village in Transylvania. You must rescue the princess from the mysterious Count van Steinoff His mansion, on the edge of the village, is replete with legend The count himself has never been seen in daylight

M30522 FDSEIDON ADVENTURE

No joysticks Poseidon Adventure is an adventure game in which you are aboard the luxury liner S S Poseidon A huge undersea earthquake hes caused a tidal wave which has capsized the ship It is floating, bottom-up on the surface and taking on water Your goal is simply to get out alive

M30533 DREAMBUG

No joysticks Dreambug is a monitor (debugger) and disassembler for the Dragon 32 which is designed to be used in conjunction with dream (editor/assembler cassette for the Dragon 32)

M30523 FINAL COUNTDDWN

No joysticks Final Countdown is an adventure game in which your mission is to prevent a nuclear missile from being launched and starting world war III You begin the game outside a missile base which has been evacuated after a berserk General has started the countdown on a missile

M30530 TIMSCRIFT

No joysticks Printer required A speed writer which will enable you to use your Dragon to produce continuous text, such as business letters, quickly and easily Timscript allows frequently used words and phrases to be replaced by two letter codes considerably reducing the number of keystrokes required The codes are automatically expanded when they are typed

L30516 DREAM

No joysticke A professional quality screen editor and assembler for use in the production of eesembly languege programe end subroutinee

I20001 ALLDREAM EDITDR/ASSEMBLER

No joysticks A screen editor, essembler, disassembler end monitor together on a cartridge The package will enable the user to create and debug assembly language programs or assembly language subroutinee to be called from within basic programs

L30520 PIXEL EDITDR
No joysticks This program will enable quick and easy creation and editing of graphics shapes Each individual pixel can be accessed in order to produce detailed pictures and character sets.

L30521 HIDE AND SEEK
No joysticks Hide and seek is a program designed by experts to aid short term memory and develop early reading skills The program has a range of difficulty levels to suit children between the ages of four and eleven, and as well as being educational, is a program which children will enjoy using

L30519 NUMBER PUZZLER
No joysticks Number puzzler is a program designed by experts to develop childrens powers of mental arithmetic The program has a range of difficulty levels to suit children between the ages of four and eleven and, as well as being educational is a program which children will enjoy

L30517 NUMBER GULPER
Joysticks optional As for number puzzler

M30526 CIRCUS ADVENTURE
No joysticks. Circus adventure is an adventure game designed especially for children between the ages of 4 and 8 The program incorporates a number of user inputs to encourage keyboard familiarity and presents the child with a series of choices to be made to encourage decision making skills The program is designed to be non-frustrating and children will enjoy using it

M30529 SCHDDL MAZE
No joysticks. As for circus adventure.

M30E10 DRAGON SELECTIDN 3
Joysticks required for aliens Dragon selection 3 is a collection of four games programs - Money, Detective. Aliens. Hangram

N30511 DRAGDN SELECTION 4
No joysticks. Dragon selection 4 is a collection of three party games for children Password, Lucky Dip, Composer

J20111 RAIL RUNNER
No joysticks A fast moving video game, rail runner is a race against time to cross a series of tracks avoiding trains and handcars to rescue Herman Hobo

M30518 EL DIABLERD
No joysticks El Diablero is an adult adventure game You are wandering in the desert trying to regain your lost magic powers before confronting El Diablero

M30527 STDRM ARRDWS
Joysticks required. Storm arrows is a fast moving multi-screen maze game Your task is to defend yourself against hostile arrows while maintaining your energy supply

K30112 GALAX ATTACK
Joysticks required As wave after wave of enemy ships attack, you must defend your ground base against them The enemy craft fly in convoy formation but they will disband in order to attack

K30532 SHARK TREASURE
Joysticks required You have discovered the lost wreck of a ship which sank many years ago with a cargo of gold bars The only thing between you and a fortune is a swarm of huge sharks Shark treasure is graphic video game

J20110 DDDDLE BUG
Joysticks required Doodle bug is a colourful, fast action maze game An increasing number of enemies have to be avoided together with a number of dangerous obstacles

K30535 SHUTTLEZAP
Joysticks required. Orbiting the earth your task is to grab as many satellites as you can before landing safely back at base. This program has great graphics and what's more it talks to you

K30114 WHIRLYBIRD RUN
Joysticks required Whirlybird run is a fast moving arcade type game, with multiple screens different types of attacker and a maze to be negotiated in the final stages

DRAGON SOFTWARE

SOFTWARE AVAILABLE FOR THE DRAGON 32

N 30500	DRAGON SELECTION ONE	Four games for the younger user Written in BASIC they can be listed and edited
N 30501	DRAGON SELECTION TWO	Collection of utilities Create your own data base write your own lunes
M 30502	QUEST	Adventure game in amedieval setting Defeat Moorlock master of the dark castle
M 30503	MADNESS AND THE MINOTAUR	A real time adult adventure game
N 30504	PERSONAL FINANCE	Keep track of family finances
N 30505	GRAPHIC ANIMATOR	Create simple cartoons on the screen and animate them by flipping through the pages Joysticks required
M 30506	COMPUTAVOICE	Your Dragon will talk with this vo ce synthesizer
N 30507	EXAMPLES FROM THE MANUAL	30 examples from the programming manual
M 30508	CALIXTO ISLAND	An adventure game Return the hidden treasure lo its nghtful place
M 30509	BLACK SANCTUM	An adventure game Overcome the forces of blach magic
M 30512	TYPING TUTOR	Improve your speach and accuracy
N 30513	DRAGON MOUNTAIN	An adventure game Defeat the guardians of the treasure hidden in the mountain
M 30514	FLAG	Race your opponent through a constantly changing maze to the final flag Joysticks required
M 30518	EL DIABLERO	An adventure game set in the desert
J.20100	BERSERK	A challenging shooting game based on the popular arcade game one or two players A high resolution game in black and white Joysticks required
J.20101	METEOROIDS	Guide your ship through treacherous asteroid belt A game requiring skill fast reactions and concentration A high resolution game in black and white Joysticks optional
J 20102	COSMIC INVADERS	Dragon version of the famous arcade game
J.20103	GHOST ATTACK	Maze game for one player Joysticks required
J 20104	CAVE HUNTER	Descend into the maze of caves in search of gold Joysticks required
J 20106	STARSHIP CHAMELEON	Protect your planet from the attacks of the Gabulators High quality arcade game with superb graphics and sound Joysticks required
J 20107	ASTROBLAST	Defend your ship against waves of attackers A high resolution game in black and white Joysticks required
J.20108	CHESS	Nine levels of play from beginner to master.
J.20111	RAIL RUNNER	Move Bill Switchman across the tracks, avoiding trains to rescue Herman Hobo

Published by Welbeck Public Relations 2 Endell Street London WC2 Printed by Drydens Printers Ltd Brent Crescent North Circular Road London NW10